1998 IEEE Workshop on Emerging Technologies, Intelligent Measurement
and Virtual Systems for Instrumentation and Measurement - ETIMVIS'98
St. Paul, MN, USA, May 15-16, 1998

A Neuro-Fuzzy Model for Missing Data Reconstruction

Simon Petit-Renaud and Thierry Denœux
Université de Technologie de Compiègne
HeuDiaSyc-UMR C.N.R.S. 6599
BP 20529, 60205 Compiègne, France
phone: 33 (0)3 44 23 44 96, fax: 33 (0)3 44 23 44 77
e-mail: spetit,tdenoeux@hds.utc.fr

A fuzzy-neuro method for reconstruction of missing data is described. This method allows to express knowledge acquired from input-output data in the form of fuzzy linguistic IF-THEN rules. The construction of these rules is based on the relations between the components of reference vectors in the measurement space. The method provides estimates of all the missing components of a vector, given an arbitrary number of available variables. An example taken from an environmental monitoring application demonstrates the effectiveness of this approach.

1 Introduction

In real applications, classical models from statistics, pattern recognition or control theory are not always well adapted to the actual knowledge of the studied system. For instance, the case often occurs where some particularly informative parameters are not available at a given time. In this paper, we propose a new technique for the *reconstruction* of missing values in a measurement vector. This approach is based on fuzzy models, which are known for their capability to express vague relations between the variables of a system [13, 12]. Unlike classical regression methods, our approach provides an estimation of *any* number of missing components in a vector, using all available ones.

In order to provide an analytical description of the relationship between the inputs and the outputs of a system, many methods have been proposed, combining fuzzy models and neural networks [8, 11]. The representation of fuzzy models as neural networks allows to apply powerful learning techniques for identifying the parameters from data [1, 2, 10]. Neural fuzzy systems combine the advantages of both neural networks and fuzzy systems: they are universal approximators [3] and they can be trained by back propagation; their parameters and structure have strong physical meaning and *a priori* knowledge can be effectively incorporated in their initial parameter values. For these reasons, this fuzzy-neuro approach has been adopted in our model.

2 Fuzzy system models

One of the main directions in the theory of fuzzy systems is the linguistic approach initially introduced by Zadeh [13]. A Linguistic Model describes a system by means of a set of IF-THEN rules with vague predicates. The general model of a fuzzy system is based on a collection of linguistic rules:

$$\text{IF} \quad x_1 \text{ is } B_{i,1} \ldots \text{ and } \ldots x_r \text{ is } B_{i,r} \quad \text{THEN} \quad y \text{ is } D_i, \tag{1}$$

where $x_1, \ldots, x_r$ are the input variables of the base sets $X_1, \ldots, X_r$, and y is the output variable taking values in Y. $B_{i,j}$ and D_i are linguistic terms of the base sets X_j and Y, respectively. These fuzzy sets are characterized by their membership functions $\mu_{B_{i,j}}$, and μ_{D_i}. They constitute the parameters of the model. The number of rules determines the structure of the linguistic model. Two alternative reasoning mechanisms may be employed with such models. We will focus on the constructive method [7], in which the aggregation of the rules is made by a union of fuzzy relations. There are also many different choices concerning the operators in the fuzzy inference system [5]. In order to have a differentiable expression between the inputs and the outputs of the system, we chose the *sum-product* rule of inference. Using this rule, the fuzzy output of the system S is given by the following expression:

$$\mu_S(y) = \sum_{i=1}^{m} \tau_i(\boldsymbol{x})\mu_{D_i}(y) \tag{2}$$

where τ_i is the degree of firing of the i^{th} rule for the input $\boldsymbol{x} = (x_1, \ldots, x_r) \in X = X_1 \times \ldots \times X_r$.

In order to obtain a crisp output value, we must *defuzzify* the fuzzy output S of the model. The usual *Center of Area* (COA) method provides the following expression for this crisp output:

$$y = f(\boldsymbol{x}) = \frac{\int_Y u\mu_S(u)du}{\int_Y \mu_S(u)du} = \frac{\sum_{i=1}^{m} y_i^\star \tau_i \int_Y \mu_{D_i}(u)du}{\sum_{i=1}^{m} \tau_i \int_Y \mu_{D_i}(u)du}, \tag{3}$$

where $f : X \longmapsto Y$ and $y_i^\star$ is the defuzzified value of the consequent fuzzy set D_i. Thus, the mapping f, determined by (3), can be represented as a three layer-feedforward network [4] (see Section 4). This method of simplified reasoning opens the possibility of learning fuzzy models from data.

3 Application to missing data reconstruction

In this paper, we use a set of fuzzy rules to solve a specific problem: missing data reconstruction. The principle of the method is the following.

Let $\mathcal{X}$ be a training set of N vectors of r variables that may have missing components, and $C = \{c_i \in I\!R^r, i \in \{1, \ldots, m\}\}$, a set of m prototypes. The prototypes, which are supposed to summarize the information given by all the vectors, may have been created by means of any clustering method such as the c-means or fuzzy c-means algorithms (note that these methods can take incomplete vectors into account). Each vector $\boldsymbol{x} \in \mathcal{X}$ may be associated to the nearest prototype $c(\boldsymbol{x})$, according to a given distance:

$$c(\boldsymbol{x}) = \arg\min_{i \in C} ||\boldsymbol{x} - c_i||$$

For every $i \in \{1, \ldots, m\}$, we can consider the set P_i of vectors which are associated to the same vector c_i:

$$P_i = \{\boldsymbol{x} \in \mathcal{X}, c(\boldsymbol{x}) = c_i\}$$

$\mathcal{X}$ is thus partitioned in m classes $(P_i)_{i=1}^m$.

If the prototypes provide a good representation of the data, we can hope to deduce the missing components of a vector near a given prototype by means of the other components. Each prototype will then define a rule expressing the relationship between the different components of the corresponding class.

3.1 Construction of the fuzzy sets

As mentioned before, the essential elements of a fuzzy model are the linguistic terms in the antecedent and consequent parts of the rules. Here, we propose to associate a rule to each prototype or to each class. For every $i \in \{1, \ldots, m\}$, let B_{ij} be a fuzzy set representing variable j of class P_i. $B_{i,j}$ can be defined by two parameters:

- c_{ij}, the j^{th} variable of the i^{th} prototype,
- σ_{ij}, the j^{th} component of the standard deviation of class P_i,

$$\sigma_{ij}^2 = \frac{1}{|Q_{ij}|} \sum_{\boldsymbol{x} \in Q_{ij}} (x_j - c_{ij})^2$$

where $Q_{ij} \subset P_i$ represents the set of vectors of class P_i, for which parameter j is known.

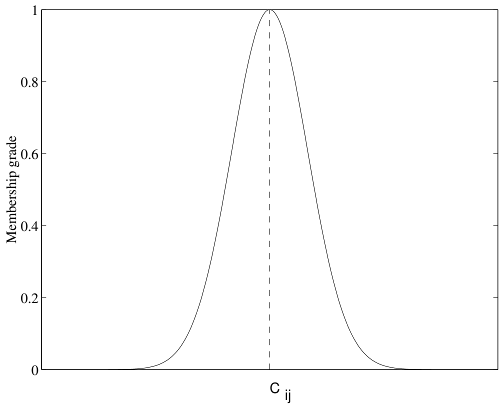

Figure 1: *Membership function of B_{ij} characterised by the two parameters c_{ij} and σ_{ij}*

The parameters of the model have a clear interpretation: roughly speaking, σ_{ij} measures the "spread" of variable j in class P_i, and the interval $[c_{ij} - \sigma_{ij}, c_{ij} + \sigma_{ij}]$ contains the "most possible" values. The shape of the membership function is more or less arbitrary. For practical reasons, we chose normalized Gaussian membership functions (Figure 1), defined by

$$\mu_{B_{ij}}(u) = \exp\left\{-\frac{1}{2}\left(\frac{u - c_{ij}}{\sigma_{ij}}\right)^2\right\}, \forall u \in I\!R. \tag{4}$$

3.2 Missing data reconstruction

The problem can now be formulated as follows. Let $\boldsymbol{x} = (x_1, \ldots, x_r) \in I\!R^r$. Let us suppose that there is a subset $I \subset \{1, \ldots, r\}$ of components which are unknown or have been declared invalid data. Let us note $V = \{1, \ldots, r\} \backslash I$. Then, for each $l \in I$ and for each prototype i, we obtain a rule $R_{i,l}$:

$$\text{IF } (\bigwedge_{j \in V} x_j \text{ is } B_{ij}) \text{ THEN } x_l \text{ is } B_{il}. \tag{5}$$

Using the sum-product rule of inference, the degree of firing τ_i of rule R_{il} is defined as:

$$\tau_i = \prod_{j \in V} \mu_{B_{ij}}(x_j) = \exp\left\{-\frac{1}{2}\sum_{j \in V}\left(\frac{x_j - c_{ij}}{\sigma_{ij}}\right)^2\right\}. \tag{6}$$

In the simplified reasoning method, the defuzzified outputs $\widehat{x_l}$ of the fuzzy system, which are estimates of the corresponding x_l, are easily obtained, according to (3), as

$$\widehat{x_l} = \frac{\sum_{i=1}^m \tau_i \sigma_{il} c_{il}}{\sum_{i=1}^m \tau_i \sigma_{il}} = \sum_{i=1}^m v_{il} c_{il} \quad \forall l \in I, \tag{7}$$

where c_{il} is the defuzzified value of the consequent part B_{il}, and $v_{il} = \tau_i \sigma_{il} / \sum_{k=1}^m \tau_k \sigma_{kl}$ is the normalized degree of firing. The estimated value of each missing component is thus a linear combination of the corresponding components of the prototypes. The weight v_{il} reflects the influence of rule R_{il} in term of distance between the considered prototype and the reconstructed pattern.

Note that, by (6) and (7), our model can be viewed as an auto-associative radial basis function neural network. At this step, we can exploit the classical advantage of neural networks: the possibility of identifying the model from training data.

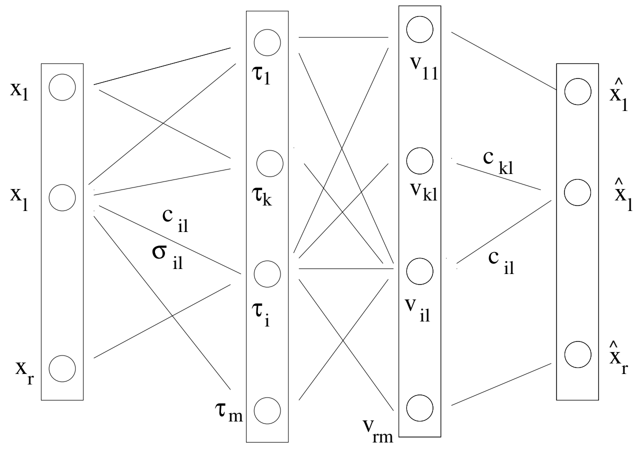

Figure 2: *Representation of the structure in a 3 layer-network*

4 Model identification

Generally, the identification of a fuzzy system model consists in two major phases: structure identification, and parameter learning. Structure identification includes the choice of input and output variables, and the determination of the form and number of rules. Here, the problem only consists in finding the best number of rules to get better generalisation ability. Many algorithms have been proposed in the literature in order to optimise the number of rules of a fuzzy system. In [9] or [6], the method is constructive. During the learning phase, a vector considered too far from the existing units will constitute a new prototype. A second way is to control the complexity of the model by adding a penalisation error to the classical Mean-Square Error. Another way, the so-called cross-validation method, consists in computing the reconstruction error of a test sample for different values of m. The chosen number corresponds to the least error. Although this method gives good results, it is computationnally demanding. Here, we propose a very simple approach, based on aggregation of neighbors and elimination of the less representative prototypes, instead of creating prototypes. Let us define the *activity rate* as the number of vectors associated to a given prototype. If the distance between two prototypes is less than a given constant ($\|c_i - c_j\| < \delta$), then they are replaced by their center of gravity c_0:

$$c_0 = \frac{|P_i|c_i + |P_j|c_j}{|P_i| + |P_j|}$$

In a second step, amongst the remaining prototypes, the ones whose activity rate is less than a given threshold are eliminated.

Now, let us focus on parameter estimation, consisting here in the determination of the σ_{ij} and c_{ij}. By (6) and (7), we have already seen that the system can be represented as an associative three-layer network (Figure 2). Thus, since the number of rules m has been determined in the preceding phase, the problem of setting up the fuzzy model becomes essentially a learning problem, which may be formalized as the minimization of a certain error function, such as:

$$e = \frac{1}{2|I|} \sum_{l \in I} (\widehat{x_l} - x_l)^2, \tag{8}$$

with respect to σ_{ij} and c_{ij}. In order to tune these parameters, the following adaptive descent method may be used. First, the initial values $c_{ij}(0)$ and $\sigma_{ij}(0)$ are chosen using the partition of the training set, as explained in Section 3.1. At each iteration t, we then compute:

$$c_{ij}(t+1) = c_{ij}(t) - \eta(t) \frac{\partial e}{\partial c_{ij}} \tag{9}$$

$$\sigma_{ij}(t+1) = \sigma_{ij}(t) - \eta(t) \frac{\partial e}{\partial \sigma_{ij}}, \tag{10}$$

Table 1: *Average test reconstruction errors for different numbers of available input variables and three different models.*

Mean error	1 input (12 models)	2 inputs (12 models)	3 inputs (4 models)
Fuzzy-Neuro Reconstruction	0.35	0.29	0.24
Nearest Prototype Reconstruction	0.61	0.42	0.35
Mean value replacement	0.41	0.41	0.41

where $\eta(t)$ is an adaptive step depending on the iteration. The error derivatives are easily obtained by straightforward calculus as:

$$\frac{\partial e}{\partial c_{ij}} = \frac{x_j - c_{ij}}{(\sigma_{ij})^2} \sum_{l \in V} (\widehat{x}_l - x_l)(c_{il} - \widehat{x}_l) v_{il} + v_{ij}(\widehat{x}_j - x_j), \tag{11}$$

$$\frac{\partial e}{\partial \sigma_{ij}} = \frac{(x_j - c_{ij})^2}{(\sigma_{ij})^3} \sum_{l \in V} (\widehat{x}_l - x_l)(c_{il} - \widehat{x}_l) v_{il} + \frac{v_{ij}}{\sigma_{ij}}(c_{ij} - \widehat{x}_j). \tag{12}$$

5 Experiments

The objective of the proposed method is to provide an estimate of any missing data, whatever the available variables. This is not a classical regression problem, since the available information varies in time. In order to demonstrate the validity of the proposed method, we considered real data from an environmental monitoring application, in which on-line measurements of four parameters (temperature, pH, conductivity and dissolved oxygen) were taken on-line in a river. A set of 700 examples was used for training, and the results were assessed using an independent test set of 700 examples, in order to estimate the generalisation capability of our method. Using the above method, the number m of rules decreased from 30 to 15. Typically, the precision of the result depends on several factors. Firstly, a good correlation between the variables is desirable, since the variables have to be reconstructed using some others. Secondly, the precision depends on the number of known variables of a vector. For a given vector, there are $r(2^{r-1} - 1) = 28$ different possible models, since for each vector $\boldsymbol{x}$, each of the r variables $\boldsymbol{x}_l$ can be reconstructed using one, two or three other variables. The estimate of $\boldsymbol{x}_l$ can have the following form:

$$\widehat{x}_l = \frac{\sum_{i=1}^{m} c_{il} \prod_{j \in V} \mu_{B_{ij}}(x_j)}{\sum_{i=1}^{m} \prod_{j \in V} \mu_{B_{ij}}(x_j)}, \quad \text{where} \quad V \in \{1, 2, 3, 4\} \backslash \{l\} \tag{13}$$

Table 1 compares the reconstruction error obtained with our method, according these kinds of models, with *two* simple reference methods. One of them is based on the replacement of missing components with the corresponding component of the nearest of the 15 prototypes. The other simply corresponds to the replacement with the mean value of the variables in the learning set. The results show that, in all the situations tested, our method seems to yield smaller errors than the other ones, even in the worst case where only one variable is known.

Figure 3 represents the outputs of the system for 200 samples in a special case, where the variable $\boldsymbol{x}_j$ is reconstructed with the three other ones. The results, obtained after 10 iterations in the optimisation phase, are compared with the nearest prototype method described above, and with our method without optimisation of the parameters. The test error was equal to 0.24 with our method after only 10 iterations, and 0.26 without any optimisation, against 0.35 with the nearest prototype method. As shown from these figures and results, prior knowledge is efficiently and immediately incorporated in the system, and the results are significantly better than those obtained with the nearest prototype method. However, it can be noticed that some variables are not well reconstructed, which is probably due to insufficient correlation with the other variables.

In order to explore the influence of the number of available data in a more detailed way, we simulated different proportions of missing data in the input variables and studied the quality of the results. Table 2 compares the reconstruction errors obtained with increasing rates of missing data. The slow increase of the error when the available data become sparser tends to demonstrate the effectiveness and the robustness of our approach.

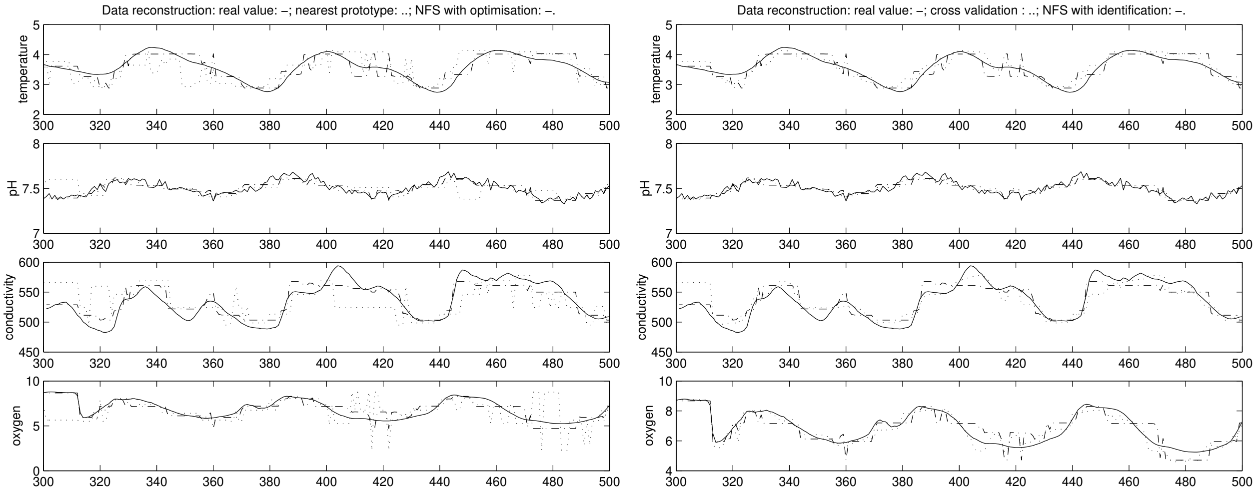

Figure 3: *Data reconstruction.* **Left***: real data (-), Neuro-Fuzzy (NF) method with optimisation (10 iterations) (-.-), nearest prototype method (..).* **Right***: real data (-), NF with optimisation (10 iterations)(-.-), NF without optimisation (..)*

Table 2: *Average test reconstruction errors for different numbers of available input variables and different proportions of missing data.*

Rate of missing values	1 input (12 models)	2 inputs (12 models)	3 inputs (4 models)
Fuzzy-Neuro Reconstruction: 0%	0.35	0.29	0.24
10%	0.36	0.30	0.26
20%	0.36	0.32	0.28
30%	0.37	0.33	0.29
40%	0.37	0.34	0.31
50%	0.38	0.35	0.33
60%	0.38	0.36	0.34
Mean value replacement	0.41	0.41	0.41

6 Conclusion

A fuzzy-neuro system for missing data reconstruction has been described. This method expresses the knowledge acquired from input-output data in the form of fuzzy inference rules, and yields estimates of all the missing components of a vector, given any number of available variables.

Acknowledgements

This work was partially supported by EEC funded Esprit Project 22442 (Environmental Monitoring and Management Systems).

References

[1] D. H. Bossley, *Neurofuzzy modelling approaches in system identification*, PhD thesis, University of Southampton, 1997.

[2] M. Brown and C. Harris, *Neurofuzzy adaptive modelling and control*, Hemel-Hampstead: Prentice Hall, 1994.

[3] Hornik, K and Stinchcombe, M. and White, "Multilayer feedforward networks are universel approximators", *Neural Networks*, vol. 2, pp. 359-366, 1989.

[4] J. S.R. Jang and C. T. Sun, "Functional equivalence between radial basis function networks and fuzzy inference systems", *IEEE Transactions on Neural Networks*, vol. 4, pp. 156-159, 1993.

[5] Lee, C.C., "Fuzzy logic in control systems: fuzzy logic controller, part 2", *IEEE Transactions on Systems, Man and Cybernetics*, vol. 20, pp. 419-435, 1990.

[6] Lee, S. and Kil, R. M., "A Gaussian potential function network with hierarchically self-organizing learning", *Neural Networks*, vol. 4, pp. 207-224, 1991.

[7] E. H. Mamdani, "Twenty years of fuzzy control: Experience gained and lessons learnt", in *Proceedings of the 2nd International IEEE Conference on Fuzzy Systems*, San Francisco, CA, 1993, pp. 339-344.

[8] H. Nomura and I. Hayashi and N. Wakami, "A learning method of fuzzy inference rules by descent method", in *Proceedings of the 1st IEEE International Conference on Fuzzy Systems*, San Diego, CA, 1992, pp. 203-210.

[9] Platt, J. C, 'Learning by combining memorization and gradient descent", in *Advances in Neural Information Processing Systems, 3: Proceedings of the 1990 Conference*. San Mateo, CA: Morgan Kaufmann, 1991, pp. 714-720.

[10] L. X. Wang, *Adaptive fuzzy system and control: design and stability analysis*, Englewood Cliffs: Prentice Hall, 1994.

[11] L. X. Wang and J. M. Mendel, "Back-propagation fuzzy system as nonlinear dynamic system identifiers", in *Proceedings of the 1st IEEE International Conference on Fuzzy Systems*, San Diego, CA, 1992, pp. 1409-1416.

[12] R. R. Yager and D. P. Filev, *Essential of fuzzy modeling and control*, New York: John Wiley and sons, 1994.

[13] L. Zadeh, "Outline of a new approach to the analysis of complex systems and decision processes", *IEEE Transactions on Systems, Man and Cybernetics*, vol. 3, pp. 28-44, 1973.